Deliverable D3.3:

SEEK as the assimilation kernel in the NEMOVAR framework

Bénédicte Lemieux-Dudon
CNRS/LEGI, Grenoble

Jean-Michel Brankard
CNRS/LEGI, Grenoble

Arthur Vidard
INRIA/LJK, Grenoble

VODA: ANR-08-COSI-016

Contents

1 Introduction

Data assimilation (DA) is divided in two main branches: (i) variational assimilation and (ii) stochastic assimilation. Unlike variational assimilation, stochastic assimilation does not rely on a functional minimization with iterative process and gradient estimations. The stochastic branch includes Kalman Filter-type approaches. Developments in both branches have been made and hybrid techniques are investigated (Krysta et al, in prep.). Till now in the field of oceanography, the 4D-Var strategy is mainly implemented for academic and research projects, and operational assimilation rather lean on ensemble and/or square root Kalman filters, possibly because the implementation of numerical linear tangent and adjoint models requires man power, time and expertise.

The VODA project focuses on variational DA. It is directly in line with the NEMO framework (Nucleus for European Modelling of the Ocean) and the NEMOVAR initiative. The NEMO framework heads the developments of the national ocean model NEMO (following-up the OPA direct code version 9) while the NEMOVAR initiative handles the project of a multi-incremental variational DA platform dedicated to NEMO (NEMOVAR is the continuation of OPAVAR, the variational DA tool that was developed for OPA version 8.2). The VODA project actively contributes to the NEMOVAR initiative with the development of the multi-incremental 4D-Var, the Linear Tangent and Adjoint models for NEMO (NEMOTAM part of the project, see deliverables D1.X). The VODA proposal and actions recently results in the first release of a TAM version phased with NEMO version 3.0, and in new NEMOVAR capabilities.

This organization is part of the NEMO community strategy which aims at delivering variational AD tools (4D-Var, 3DFgat,...) specifically dedicated to NEMO and in phase with the latest releases. Beyond variation AD tools, the community might expect to benefit from a larger panel of DA tools. The VODA task 3.3 is a preliminary step in this direction.

The objectives of task 3.3 are to show the feasibility of hosting non-variational DA tools, in the new NEMO/NEMOVAR structure. When rebuilding the incremental 4D-Var from OPAVAR, the coding, building and execution operations have been reorganized in three independent components NEMO and NEMOVAR, with a clever distribution of the DA routines and an off-line communication. This organization is a key point for task 3.3 (see details section 2.1). The sequential AD tool SESAM that is designed to build experiments of square root Kalman Filter type was chosen for that demonstration.

The work related to task 3.3 consisted in bringing together NEMO and SESAM in the NEMO/NEMOVAR structure, in order to perform SEEK (Singular Evolutive Extended Kalman) experiments. A python script has been developed for that purpose. It enables to

drive demos and/or user-defined SEEK experiments. This document provides guidelines to apply the python driver. It also provides technical details for developers in charge of future developments.

In section 2, we make a short description of NEMO and SESAM specificities that were determining to decide on the way to organize communication and operations between the two executables. In section 3, we present the python demonstrator, its capacities, guidelines to apply it and the demo results. In section 4, we explain how to run a user-defined SEEK experiment with the python driver. The last two sections, are dedicated to developers and provide more information on the NEMO and SESAM codes. Conclusion/perspectives

2 NEMO and SESAM in short

2.1 NEMO/NEMOVAR structure

Data assimilation applied to ocean models usually consists in improving the forecast model trajectory x_i^f by correcting the model initial condition x_0^b (called background) with information brought by observations y_i^o. All DA methods (variational or stochastic) use the misfit between observations and model predictions to search the correction δx_0^a (called analysis increment), and then apply it to the background to provide the analysed state vector $x_0^a = x_0^b + \delta x_0^a$. On the opposite, operations that are required to compute the increment of analysis are specific to the type of assimilation kernel.

The NEMO/NEMOVAR structuring is based on this distinction. DA routines non-specific to the variational assimilation kernel have been included in NEMO, while DA routines strictly specific to the variational kernel remains in the NEMOVAR component. In reference to the incremental 4D-Var algorithm, which is organized with two loops called inner and outer loops, the NEMO executable was re-baptised *outer loop* in the VODA project, because it now has all the capability to perform the 4D-Var outer loop operations: (i) it computes the model equivalent and innovation vectors, (ii) it applies the increment of analysis to the background. NEMOVAR to which the *inner loop* executable is associated, only operates DA routines that are strictly specific to the variational kernel [1].

This structure renders easy the application of different assimilation kernel: the NEMOVAR *inner loop* must simply be replaced by another kernel which computes the increment of analysis. This work shows a demonstration of such manipulation with the SESAM sequential assimilation kernel.

2.2 SESAM specificities

SESAM is developed and maintained by the MEOM team at the LEGI laboratory. It is designed to perform sequential data assimilation of Square Root or ensemble filter type, as

[1] The *inner loop* enables to apply a 4D-Var incremental analysis: (i) it builds an approximation of the misfit function by linearising the model, (ii) it minimizes the resulting quadratic misfit function on the basis of the gradient, which involves the adjoint operator.

well as a variety of connected operations. Information can be found on the MEOM webpage http://www-meom.hmg.inpg.fr/Web/Outils/SESAM/sesam.html. The modularity of the SESAM code enables to call elementary operations directly at the prompt of a shell, by specifying a SESAM module name with possibly a set of module arguments as well as general options:

`sesam [- options] -mode module_name module_arguments`

SESAM modules are classified in three groups: (1) Analysis, (2) Diagnosis and (3) Utility. The Analysis group essentially includes capabilities related to the Reduced Rank Square Root Kalman Filters. The Diagnosis group enables statistical analysis with especially EOF (Empirical Orthogonal Functions) treatment. The Utility group helps to perform arithmetic operations, file conversions and database extractions. The table 1, presents the modules that have been applied in the present work.

Category	Module	Description
Analysis	groa	Global reduced order analysis
	lroa	localized reduced order analysis
Diagnosis	geof	Global EOF decomposition
	diff	Computes RMS difference
Utility	intf	Interface between file format associated to data type
	zone	Generates domain partition and bubbles of influence for localization
	obsv	Extraction of observations from databases

Table 1: SESAM modules applied in this work: a more extensive list can be found on the SESAM website http://www-meom.hmg.inpg.fr/Web/Outils/SESAM/sesam.html.

Apart from modularity and interactivity, SESAM is an off-line tool disconnected from any particular model. The communication between SESAM and the model on which it applies, is operated by the mean of files. By itself SESAM can be processed with any executable code producing a model trajectory[2]. This file-based communication applies to all sort of SESAM elementary operations and requires well-defined naming conventions for input or output data files or directories. These naming rules enable SESAM to recognize on one hand the file format but most importantly the type of objects stored in the file. Details on naming convention are provided in appendix B.1 section C.

2.3 NEMO and SESAM: building a SEEK experiment

2.3.1 SEEK and Kalman filter: short recall

The SEEK is a particular type of Kalman Filter (Singular Evolutive Extended Kalman). It is: (i) an extended filter (non-linear model), of (ii) reduced rank (only few principal modes

[2]Korn shell scripts are available upon request to perform Kalman assimilation on both OPA and Hycom ocean models.

of the background error covariance matrix are considered, which comes up to a reduced rank matrix), and of (iii) square root type (the principal modes are handled instead of the covariance matrix itself). It is organized in the three classical Kalman filter steps (where index $k = 0, 1, \ldots$ refers to discrete time, and superscripts b, f and a refers to background, forecast and analysis respectively):

1. **Initialization** of:
   - x_0^b the background state vector (a priori estimate of the initial state of the model),
   - B the background error covariance matrix.
2. **Analysis** where are estimated the Kalman optimal gain K_0^* and subsequently:
   - x_0^a the analysed state vector or equivalently $\delta x_0^a = x_0^a - x_0^b$ the increment of analysis,
   - P^a the analysed error covariance matrix.
3. **Forecast** applied on the assimilation time window $[t_0, t_q]$ to estimate at time steps $t_0, t_1, \ldots, t_q$:
   - x_k^f the trajectory of the state vector,
   - P_k^f the forecast error covariance matrix.

In the case of Extended Kalman Filters the dynamic model and/or observation operators are non-linear:

- the dynamic model $\mathcal{M}$ transforms x_k into x_{k+1}, the state vectors at time t_k and t_{k+1} respectively (see appendix B for a more rigorous definition). The linear tangent operators $\mathbf{M}_k$ calculated at time t_k is given by:

$$\mathbf{M}_k = \left.\frac{\partial \mathcal{M}}{\partial x}\right|_{x_k} \tag{1}$$

- the observation model $\mathcal{H}$ predicts observation y_k^o, by calculating the model equivalent from the state x_k^f. The linear tangent operators $\mathbf{H}_k$ is given by:

$$\mathbf{H}_k = \left.\frac{\partial \mathcal{H}}{\partial x}\right|_{x_k} \tag{2}$$

Setting t_k as the time of analysis, the Kalman filter equations are:

1. **Initialization**:

$$\begin{array}{ll} x_0^f & = x_0^b \\ P_0^f & = B \end{array} \tag{3}$$

2. **Analysis**:

$$\begin{aligned} K_k^* &= P_k^f \, {\mathbf{H}_k}^T (\mathbf{H}_k \, P_k^f \, {\mathbf{H}_k}^T + R_k)^{-1} \\ &= ({P_k^f}^{-1} + {\mathbf{H}_k}^T \, {R_k}^{-1} \, \mathbf{H}_k)^{-1} {\mathbf{H}_k}^T \, {R_k}^{-1} \end{aligned} \tag{4}$$

$$\begin{aligned} x_k^a &= x_k^f + K_k^* \left(y_k^o - \mathcal{H}_k(x_k^f) \right) \\ P_k^a &= (I - K_k^* \, \mathbf{H}_k) \, P_k^f \end{aligned} \tag{5}$$

3. **Forecast**:

$$\begin{aligned} x_{k+1}^f &= \mathcal{M}_{k,k+1}(x_k^a) \\ P_{k+1}^f &= \mathbf{M}_k \, P_k^f \, {\mathbf{M}_k}^T + Q_k \end{aligned} \tag{6}$$

where Q_k is the dynamic model error covariance matrix (see appendix B for details).

One of the SEEK classical hypothesis states that background error modes are correctly represented by the model directions of largest variability. According to this hypothesis, the background error covariance matrix B is initialized by applying a Principal Component Analysis (PCA), also referred as Experimental Orthogonal Functions (EOF), on a given ensemble of state vectors along the model trajectory (see appendix A).

2.3.2 Task distribution between NEMO and SESAM

Task 3.3 (detailed in the introduction 1) consisted in the replacement of the NEMOVAR *inner loop* executable by a sequence of calls to some specific SESAM modules (see table 1) to perform a SEEK experiment in the new structure NEMO/NEMOVAR. In the NEMO/NEMOVAR project, the communication between NEMO *outer loop* and NEMOVAR *inner loop* is off-line. It involves formatted input and output files. At the same time, SESAM also operates off-line with its own set of formates I/O files On this basis, it was straightforward to opt for an off-line communication between NEMO *outer loop* and SESAM. An important part of the work therefore consisted in file format conversions. The other part of the work consisted in controlling the NEMO and SESAM formatted parameter files (i.e., namelist, sesamlist, ...etc.). In appendix B.1 section C, are described the main I/O files at hand and the connected operations of file format conversion.

Hereafter is detailed the way computing tasks are organized between NEMO and SESAM, on one hand for the **Initialization of the B matrix**, which is synthesized on figure 1 and on the other hand for the **SEEK cycle** itself, which is shown on figure 2. The organization goes as follows:

1. **Initialization of x_0^f and P_0^f** performed both by NEMO and SESAM:
   - NEMO provides $x_0^f = x^b$ the initial background state vector;
   - NEMO provides a set of state vectors x_j along the model trajectory, in order to build $\mathcal{X}$, the data matrix, and $\mathcal{X}\mathcal{X}^T$, the associated covariance matrix, which is used to approximate $P_0^f = B$, the background error covariance matrix (see appendix A);

- SESAM performs the PCA analysis by applying a singular decomposition to the data matrix $\mathcal{X}$, which is equivalent to apply a principal mode decomposition to $\mathcal{X}$ (see appendix A). SESAM selects r principal modes (on the higher variance criterion), which enables to build S_0^f a $n \times r$ matrix, which is one of the B square root matrices of rank r:

$$B = S_0^f \, S_0^{f\,T}$$

2. **Cycles of analysis applied to the background state vector** x_k^f:

- NEMO provides the model equivalent $H\,x_k^f$, which is the model prediction of observation y_k^o, at the right time t_k;

- SESAM computes K_k^* and subsequently δx_k^a and S_k^a a reduced rank square root of P_k^a, the error covariance matrix of analysis;

3. **Forecast**:

- NEMO applies the increment of analysis δx_k^a to the background and calculates the forecast trajectory x_{k+1}^f; a fix basis algorithm is applied, which does not propagate S_k^a with the model, so that $S_{k+1}^f = S_k^a$.

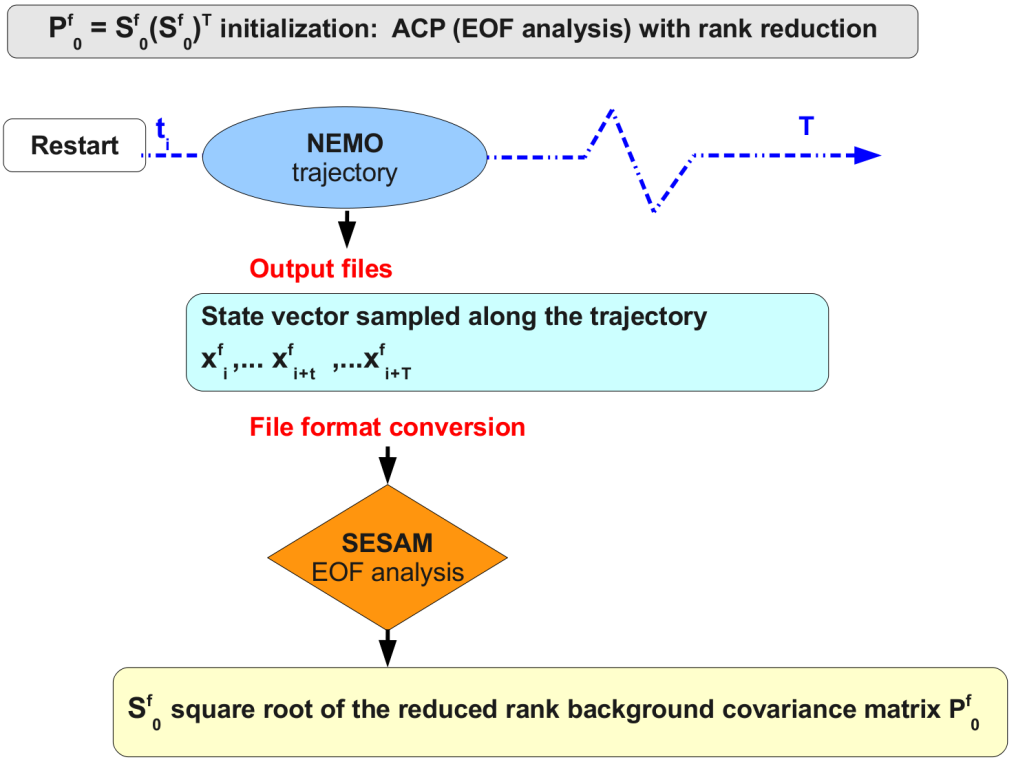


Figure 1: **Organization of the ACP analysis tasks between NEMO and SESAM** The background error covariance matrix of reduced rank S^f_k is built by the SESAM *geof* module, on the basis of an ensemble of state vectors sampled along the NEMO model trajectory. The NEMO trajectory files are of type tam_trajectory*.nc type and previously converted to SESAM file format.

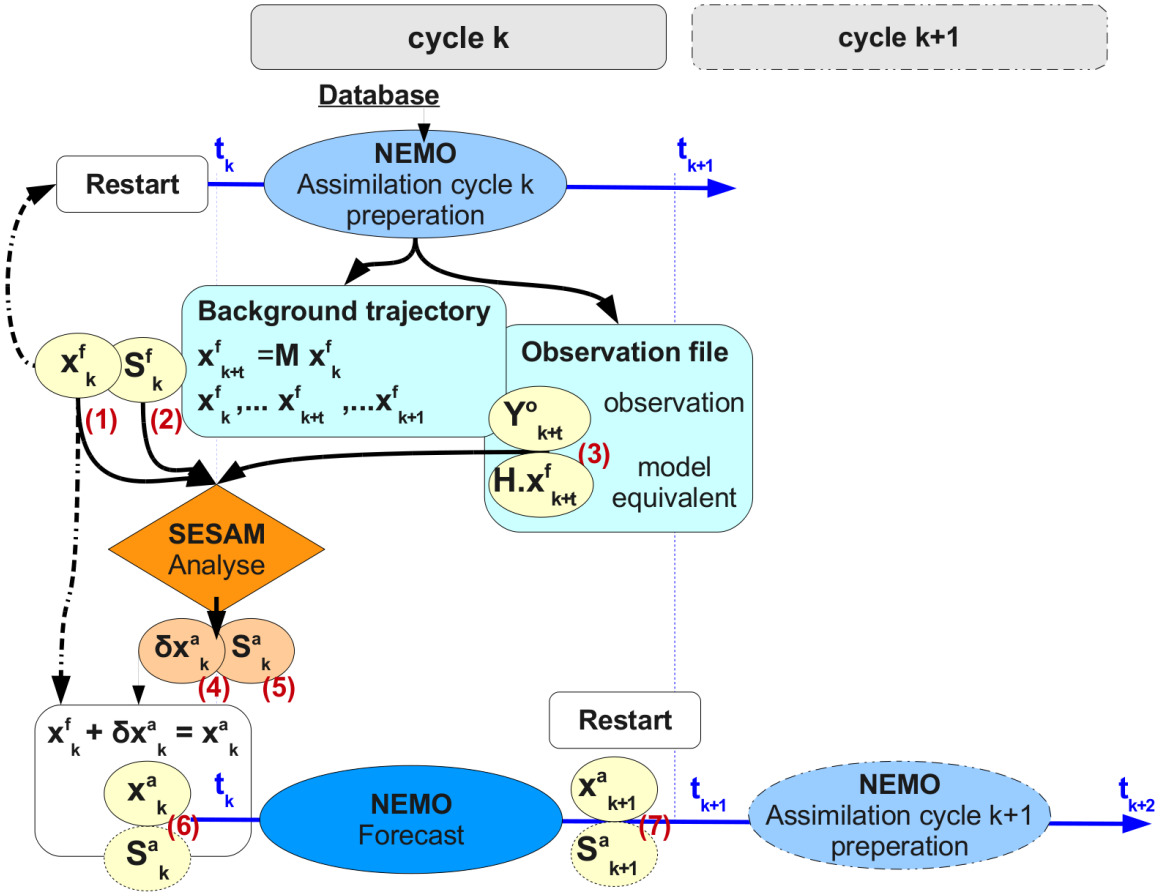


Figure 2: **SEEK cycle operations between NEMO and SESAM** Each assimilation cycle is prepared with a Nemo run called *NEMO assimilation cycle preparation*, followed by the *SESAM analysis* and the *NEMO forecast* steps. The *NEMO assimilation cycle preparation* provides input files required both: (i) by the *SESAM analysis*, and (ii) by the *NEMO forecast*. In particular, it reads the observation y_t^o in a database, calculates the model equivalent $\mathbf{H}_k x_t^f$ with the *NEMO observation operation module*, and stores both quantities in a *NEMO feedback file* (label 3). It moreover stores the background state vector: (i) first in a file called tam_trajectory*.nc that will be further converted in SESAM file format (label 1) for the analysis, and (ii) second in a file called assim_background_IAU/DI.nc, which will be combined to the analysis increment (label 4) at the beginning of the NEMO forecast step. The feedback (observation) and trajectory (x_0^f) files are converted to SESAM file format before the analysis performed with modules *groa* or *lroa*. The background error covariance matrix of reduced rank S_k^f previously built by SESAM is stored as files in a special directory. The *SESAM analysis* computes the analysis increment and error covariance matrix and stores them in files (labels 4 and 5). Then, a fix basis algorithm is applied: only the analysis increment file is converted in a NEMO file called assim_background_increment.nc. The *NEMO forcast* run then combined it with the assim_background_IAU/DI.nc file (label 6), and further computes the analysed trajectory that will provide the background state (a restart file) to the next assimilation cycle.

For programmers, a sketch is provided in appendix B.1 section D.2, to show in detail the NEMO and SESAM files involved in a SEEK cycle.

3 Demonstrator

3.1 Python script to drive the demonstrator

To organize the sequence of calls to SESAM and NEMO and the communication between both executables, a python driver was written. Its name is currently `seq_assim_driver.py` and it can be found in the VODA project directory UTIL/python/piano/sesam. Five demo experiments can be ran by calling the script at the prompt, with one of the following demo experiment name as argument:

- **SQB-SEEK/trjonly** which builds an ensemble of m state vectors $X(r_i, t_j)$ along the model trajectory, where t_j and r_i refer to the time an model space grids, with indices j and i such as $j = 1, \ldots, m$ and $i = 1, \ldots, n$.
- **SQB-SEEK/rrkonly** which applies a PCA analysis on a user-defined data matrix $\mathcal{X}$, built with a given ensemble of state vectors $\{X(r, t_j), \quad \forall\, j = 1, \ldots, m\}$, and with a user-defined rank reduction r.
- **SQB-SEEK/trj2rrk** which successively applies experiments **trjonly** and **rrkonly**.
- **SQB-SEEK/seq_asm_sgl_groa** which consists in 2 SEEK cycles in the framework of a single observation.
- **SQB-SEEK/seq_asm_sgl_lroa** which consists in 2 SEEK cycles with localization in the framework of a single observation.
- **SQB-SEEK/seq_asm_prf_lroa** which consists in 2 SEEK cycles with localization and a real ENACT temperature profile assimilated.

In order to launch a PCA analysis for example, the user has to type:

```
python seq_assim_driver.py  SQB-SEEK/rrkonly
```

Each demo experiment name corresponds to a directory that can be found under UTIL/python/piano and that stores several formatted files which define the experiment. This procedure is consistent with the python script which currently drives variational DA experiments (the so-called piano driver, see document http://ljk.imag.fr/membres/Claire.Chauvin/VODA/piano-reference-manual.pdf). In section 4, guide lines are provided to modify the experiment files. Compared to the variational driver, slight differences exist in the organization of the SEEK demo and experiments. All the SEEK demos are collected in the UTIL/python/piano/exp/SQB-SEEK directory, which therefore includes several demo types that are classified in subdirectories:

- *trjonly* where parameters to build the ensemble of state vectors are defined,
- *rrkonly* or *trj2rrk* where parameters to perform the PCA and rank reduction analysis are defined,
- *seq_asm_sgl_groa*, *seq_asm_sgl_lroa* and *seq_asm_prf_lroa* where parameters for the SEEK cycles are defined.

Table 2 summarize the correspondence between demo experiments, SEEK steps, script arguments,... etc.

SEEK step	**Available experiment**	**Experiment demo name**
Initialization	State vector ensemble	SQB-SEEK/trjonly
	ACP analysis	SQB-SEEK/rrkonly
	Ensemble + ACP analysis	SQB-SEEK/trj2rrk
Analysis/Forecast SO	SEEK cycles	SQB-SEEK/seq_asm_sgl_groa
	SEEK cycles with localization	SQB-SEEK/seq_asm_sgl_lroa
Analysis/Forecast RP	SEEK cycles with localization	SQB-SEEK/seq_asm_prf_lroa

Table 2: Available experiments, corresponding SEEK steps, experiment demo names to pass to the python driver `seq_assim_driver.py`, where SO and RP means Single Observation and Real Profile experiment respectively.

3.2 Prerequisites to run the demo

3.2.1 Building NEMO and SESAM

To run the SEEK demonstrator, both NEMO and SESAM must be compiled. In the present work, the fcm build system (Flexible Configuration Management, see http://www.metoffice.gov.uk/research/collaboration/fcm) was applied to the SESAM sources. This action enables to build NEMO and SESAM in a consistent manner (fcm already manages the building operation for the NEMO NEMOVAR NEMOTAM sources), by applying a common building script `fcmvmake.ksh`. The later script, which can be found in the directory UTIL/build, has to be succesively applied to NEMO and SESAM by typing at the prompt:

1. `./fcmvmake -B NEMO -g` *`GRID`* `-t` *`WRKDIR`* `-c` *`ARCH`*
2. `./fcmvmake -B NEMO -g` *`GRID`* `-t` *`WRKDIR`* `-c` *`ARCH`*

where the following arguments must be set by the user:

- *`GRID`* is the name of the ocean configuration grid,
- *`WRKDIR`* is the user-defined working directory, where the executables and output files are created during the building and simulation steps respectively,

- *ARCH* is the name of the directory in UTIL/scripts/builb/bld where can be found the nemo.cfg file, which is an fcm configuration file containing user-defined building options (e.g., compiler name and library to be linked,...etc.).

3.2.2 Software requirements

The python driver `seq_assim_driver.py` currently calls the nco tool (netCDF Operator), which handles netcdf file operations (see nco web page for further information: http://nco.sourceforge.net/)[3].

Regarding the python environment, the user may have to install two scientific packages, which are not always provided in the standard distribution: (i) Numeric Python and (ii) Scientific Python. The former provides access to the numpy module. The latter enables the use of the Scientic.IO.NetCDF module.

At last, all the requirements connected to the NEMO compilation or execution must be fullfilled to apply the task 3.3 demonstrator (e.g., fortran compiler with netcdf library installed, etc.).

3.2.3 Ocean configuration

The demonstrator was only tested on the SQB-Z11 ocean configuration. It might be extended to other configurations in the future.

3.3 Demo gallery

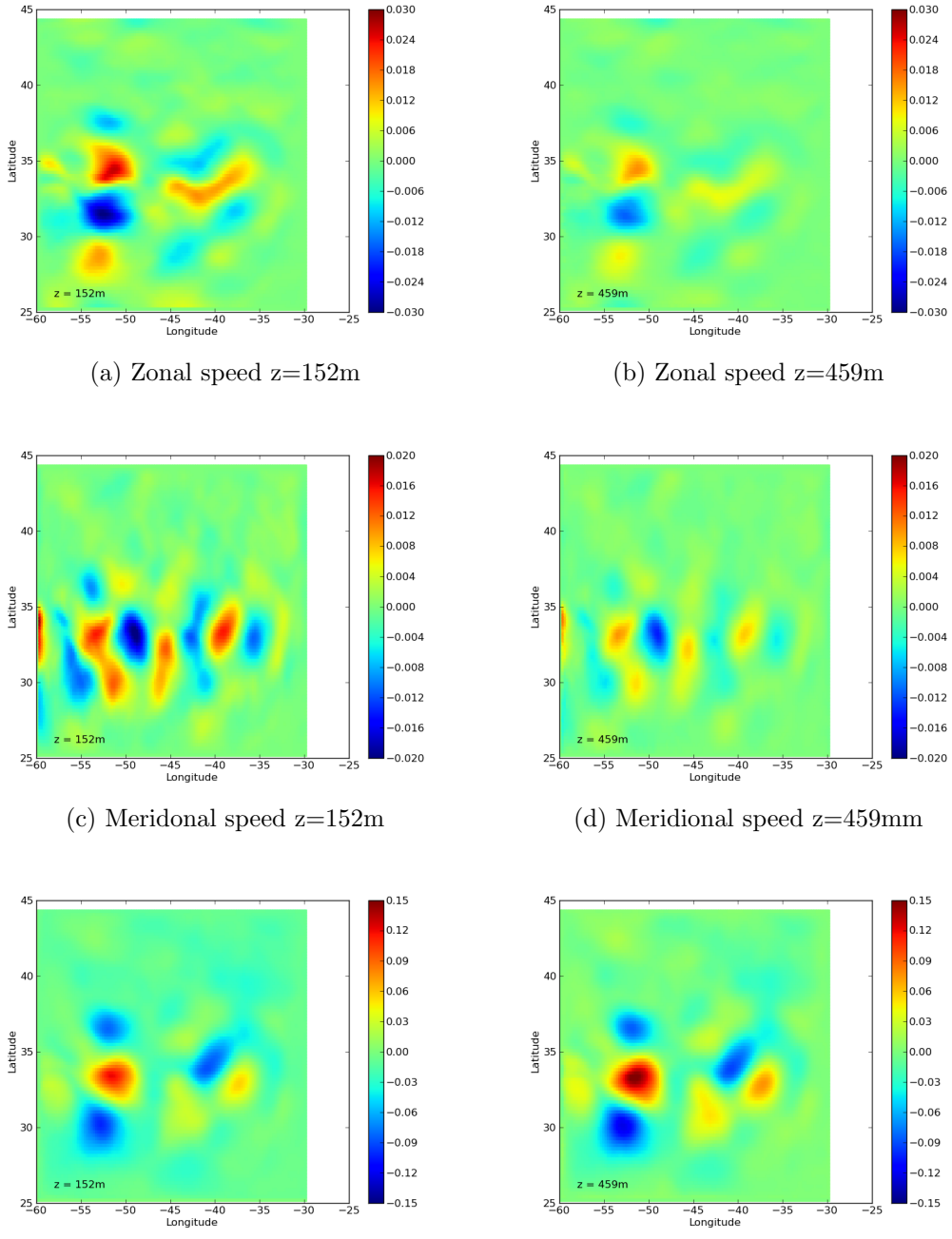
3.3.1 Initialization of the background error covariance matrix

We show hereafter some results obtained with the PCA analysis and rank reduction demos, which is to say demos *SQB-SEEK/trj2rrk* or equivalently *SQB-SEEK/trjonly* followed by *rrkonly*. The target ensemble is made of 731 members that were picked every 5 days along a 10 year model trajectory. Before the PCA analysis, the members of the ensemble have been priorly weighted to prevent the principal component to mainly represent the variables showing the larger standard deviation. The weights are therefore distinct for t, s, u, v and $sshn$ and chosen for each variable, as the inverse of their standard deviation over the ensemble that was further on space averaged on the whole domain (the variable standard deviation over the ensemble is a 3D field which depends on x, y and z coordinates).

The 5 day sampling period of the state vectors along the 10 year trajectory does not provide statistically independent members in the ensemble. For that reason, we expect only a poor representation of the long distance correlation even with no rank reduction at all. A more appropriate sample should be built with state vectors picked every month or more on a longer model trajectory; eddy events are indeed known to be time decorrelated over time period of orders of magnitude close to 2 or 3 months (J.M. Brankart, personal communication). The applied ensemble probably only contains 30 to 40 Eddy events, which is not much considering that the statistical process should capture their spatial correlation.

[3] A future objective is to replace calls to the nco by calls to netCDF python routines.

Figures 3 and 4, show the first two principal component for the temperature, the zonal and meridional velocities, at level z equals to 152 and 459 meters, respectively. Figure 5 show the same PC component for the see surface height.



(a) Zonal speed z=152m

(b) Zonal speed z=459m

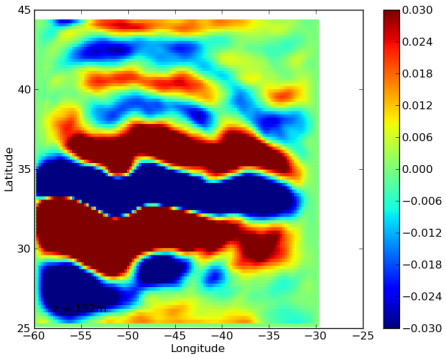
(c) Meridonal speed z=152m

(d) Meridional speed z=459mm

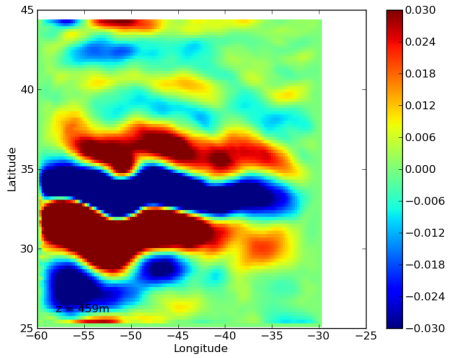
(e) Temperature z=152m

(f) Temperature z=459m

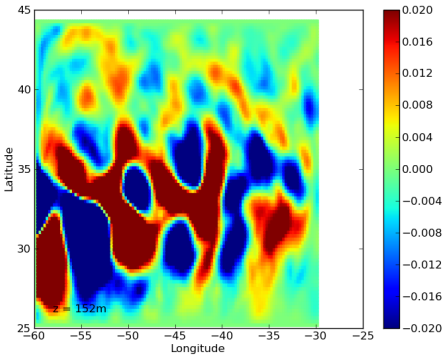
Figure 3: First principal component for u, v and t fields.



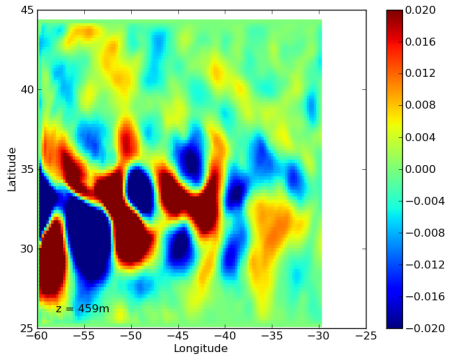
(a) Zonal speed z=152m



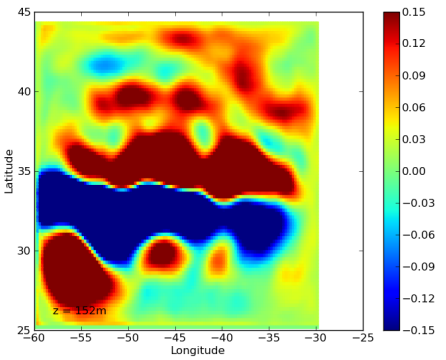
(b) Zonal speed z=459m



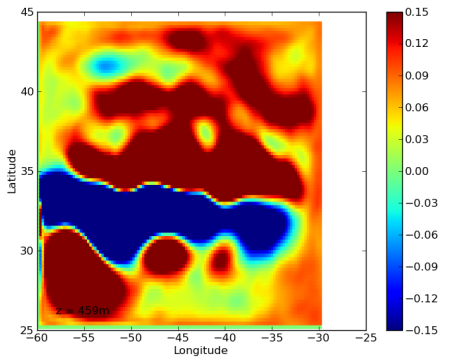
(c) Meridonal speed z=152m



(d) Meridional speed z=459m

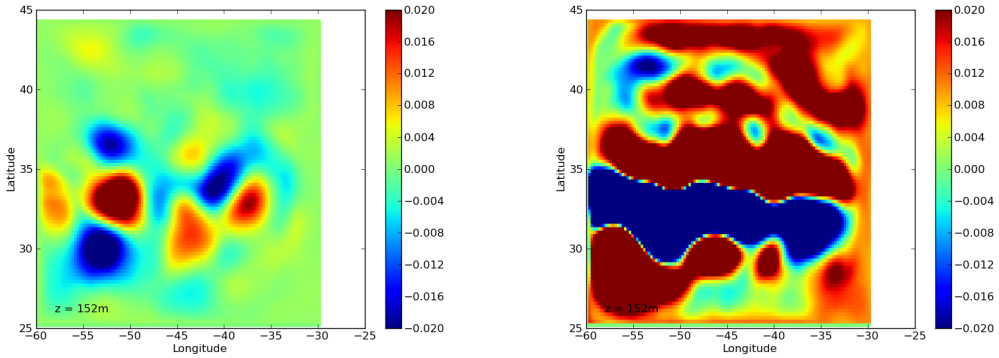


(e) Temperature z=152m



(f) Temperature z=459m

Figure 4: Second principal component for u, v and t fields.



(a) First principal component (b) Second principal component

Figure 5: First and second principal component for ssh field.

SESAM stores the analysis increment coefficients on the principal component basis. An excerpt in appendix section D.1 shows that we cannot attribute the spurious observed spatial correlation to a single given principal component.

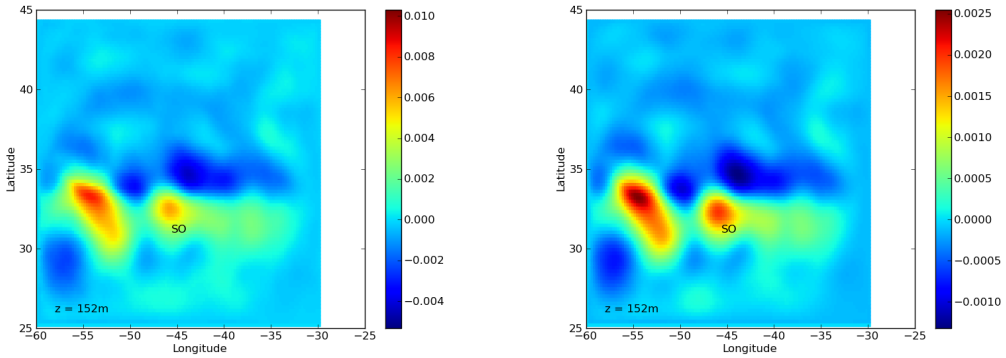
3.3.2 SEEK without localization: single observation experiment

We here present the results obtained with the demo experiment *SQB-SEEK/seq_asm_sgl_groa*: a single temperature observation is assimilated with no localization treatment. Experiments consist in 2 assimilation cycles lasting 12 days each, applied on the SQB-Z11 ocean configuration, with a fictious temperature observation localized near the top z-level of the grid, and which is the model equivalent temperature with a bias. Longitude and latitude of the observation are -45.5 and 31 °close to the SO/WE part of the jet The initialization of the B matrix is based on the ACP demo detailed in the previous section and where we have retained 30 principal components. We conducted three variants of this experiment by changing the applied temperature bias and/or the observation error:

- experiment variant 1 is plotted on figures 6 and 7:
  the bias is of 1°C, the relative error of 5%
- experiment variant 2 is plotted on figures 8 and 9):
  the bias is of 0.5°C, and the relative error of 5%
- experiment variant 3 is plotted on figures 10 and 11:
  the bias is of 1°C, the relative error of 7.5%

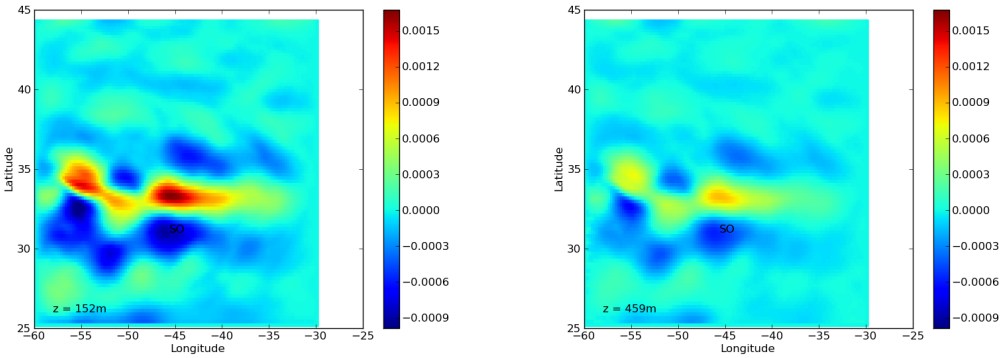
The six figures related to experiment variant 1 to 3, show the increments of analysis of fields t, u and v at vertical levels z1=152m and z2=459m, as well as the ssh field at the top z-level. For all experiment variants, we obtain the expected results. A higher bias or a lower observation error involves higher field perturbations around the observation:

the temperature is higher, the density modified, the sea surface height increased and a gyre developed itself around the perturbation. However, important field changes are also observed away from the assimilated observation, especially on the highly correlated areas of the jet. These changes are very likely due to spurious spatial correlations recorded in the S_0^f matrix, precisely because of the inadequate ensemble properties (see comments section 3.3.1).



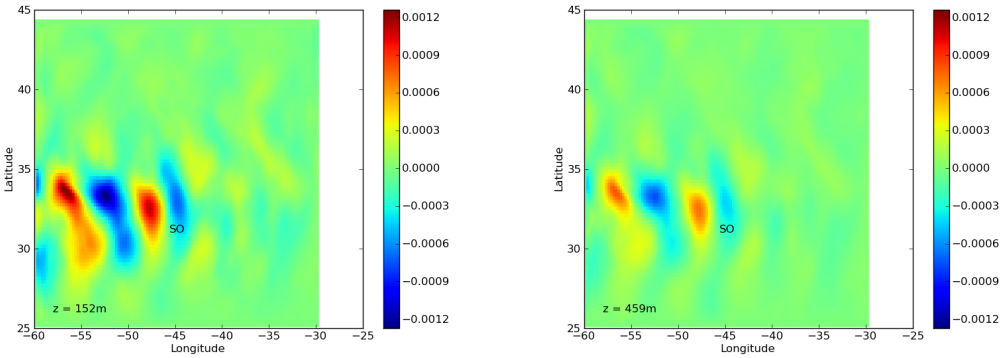
(a) Temperature z=152m (b) Sea Surface Height z=152m

Figure 6: Increment of analysis of Sea Surface Height and temperature for experiment variant 1 (bias of 1°C, relative error 5%), where SO annotation shows the localization of the Single Observation.



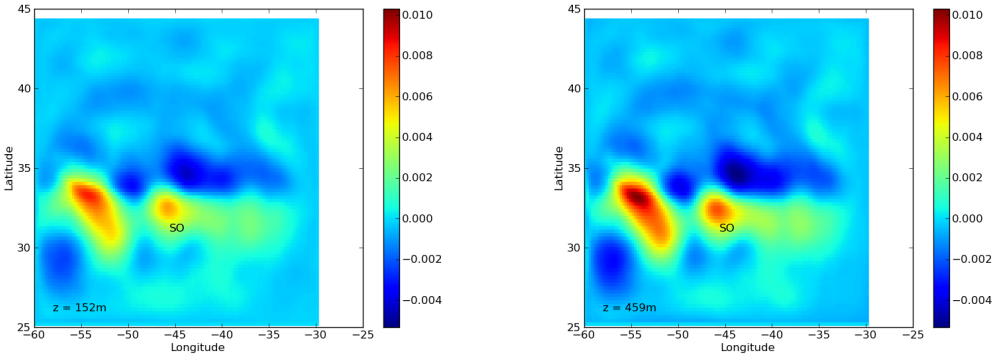
(a) Zonal speed z=152m

(b) Zonal speed z=459m



(c) Meridonal speed z=152m

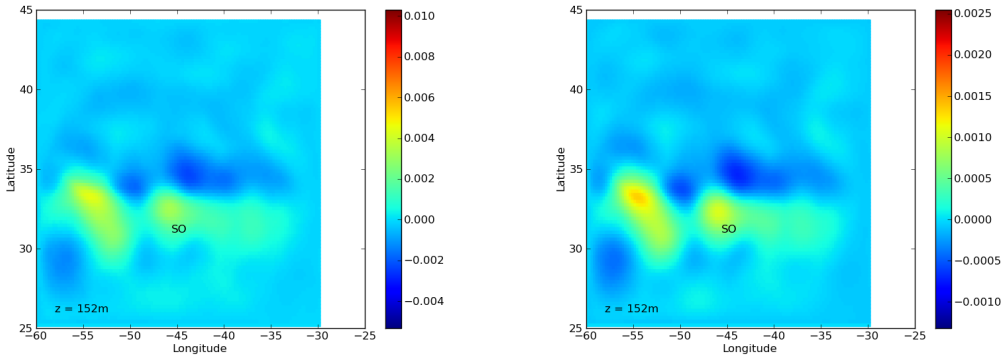
(d) Meridional speed z=459m



(e) Temperature z=152m

(f) Temperature z=459m

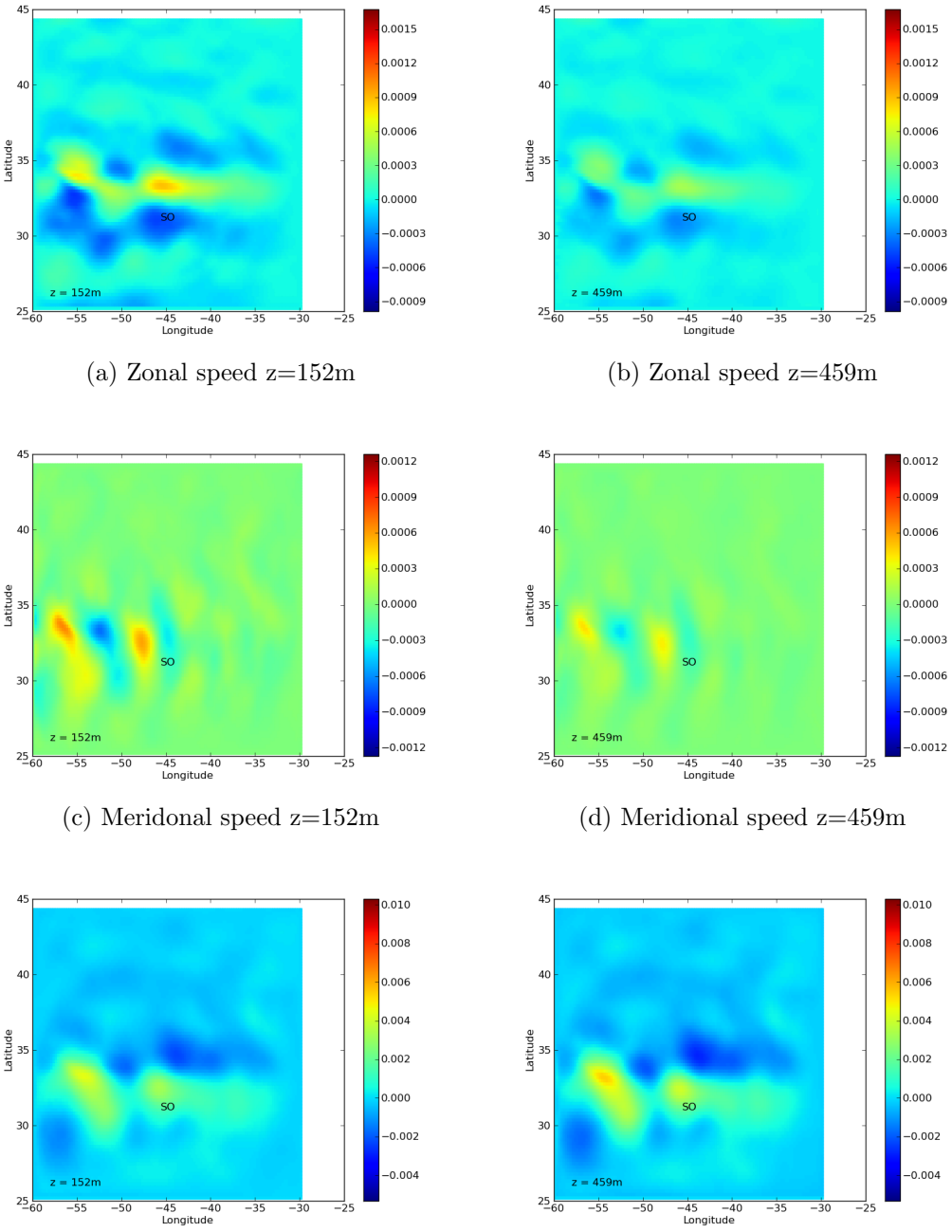
Figure 7: Increment of analysis for u, v, t fields for experiment variant 1 (bias of 1°C, relative error 5%), where SO annotation shows the localization of the Single Observation.



(a) Temperature z=152m

(b) Sea Surface Height z=152m

Figure 8: Increment of analysis of Sea Surface Height and temperature for experiment variant 2 (bias of 0.5°C, relative error 5%), where SO annotation shows the localization of the Single Observation. The colobar palet, dynamic and range are chosen equal to figure 6 for comparison.



(a) Zonal speed z=152m

(b) Zonal speed z=459m

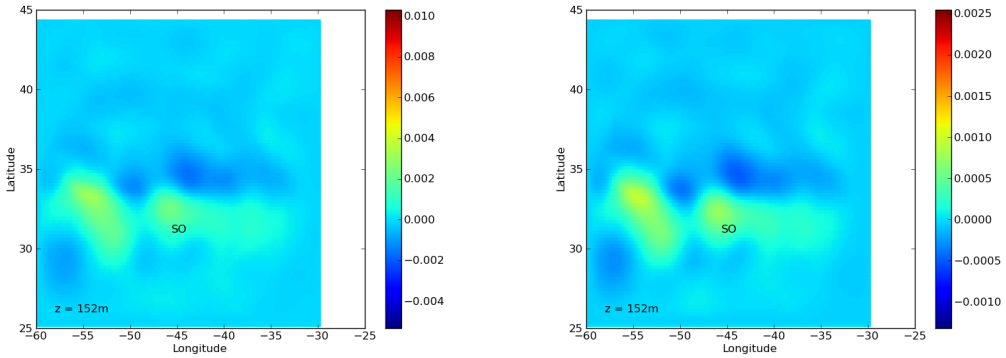
(c) Meridonal speed z=152m

(d) Meridional speed z=459m

(e) Temperature z=152m

(f) Temperature z=459m

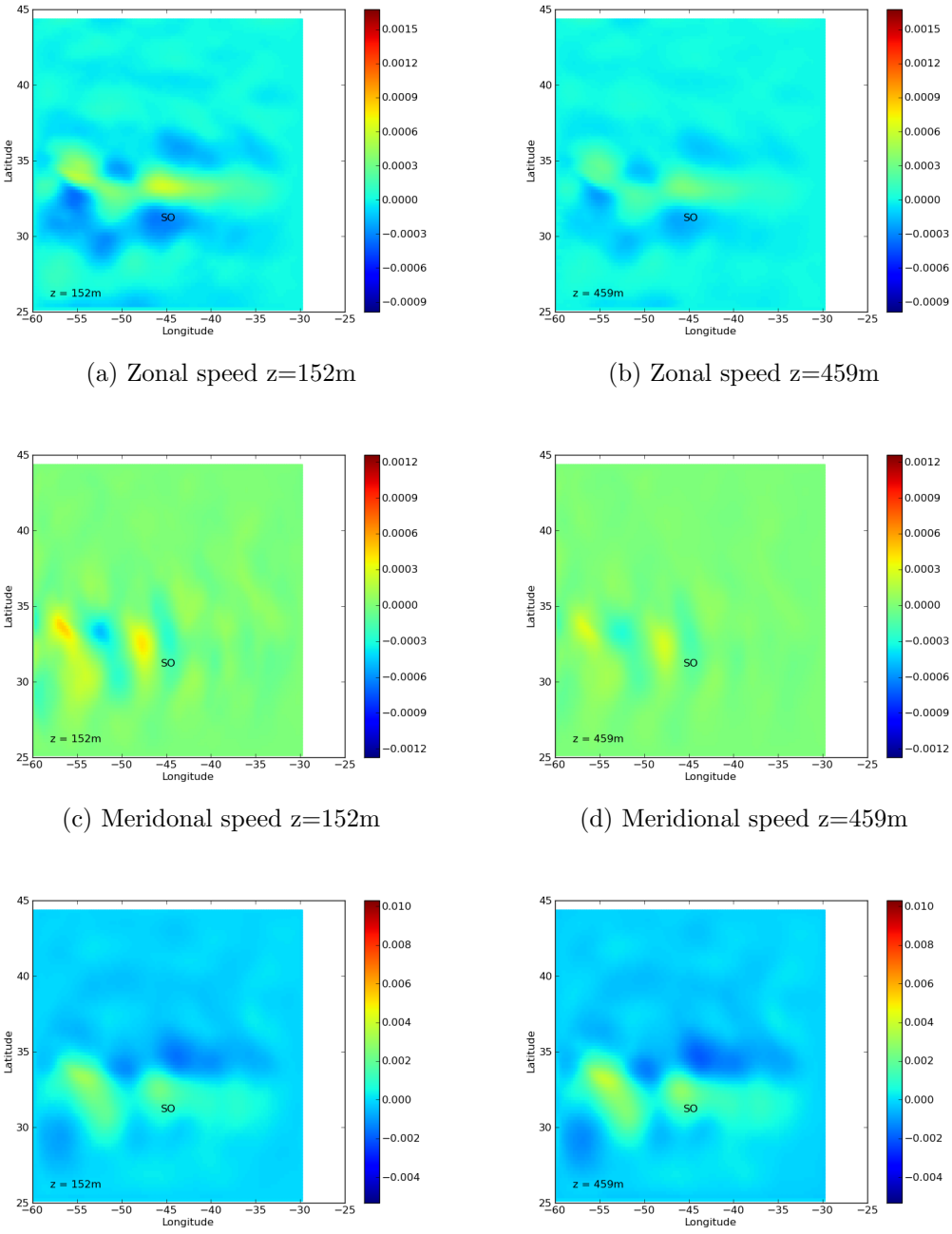
Figure 9: Increment of analysis for u, v, t fields for experiment variant 2 (bias of 0.5°C, relative error 5%), where SO annotation shows the localization of the Single Observation. The colobar palet, dynamic and range are chosen equal to figure 7 for comparison.



(a) Temperature z=152m

(b) Sea Surface Height z=152m

Figure 10: Increment of analysis of Sea Surface Height and temperature for experiment variant 3 (bias of 1°C, relative error 7.5%), where SO annotation shows the localization of the Single Observation. The colobar palet, dynamic and range are chosen equal to figure 6 for comparison.



(a) Zonal speed z=152m

(b) Zonal speed z=459m

(c) Meridonal speed z=152m

(d) Meridional speed z=459m

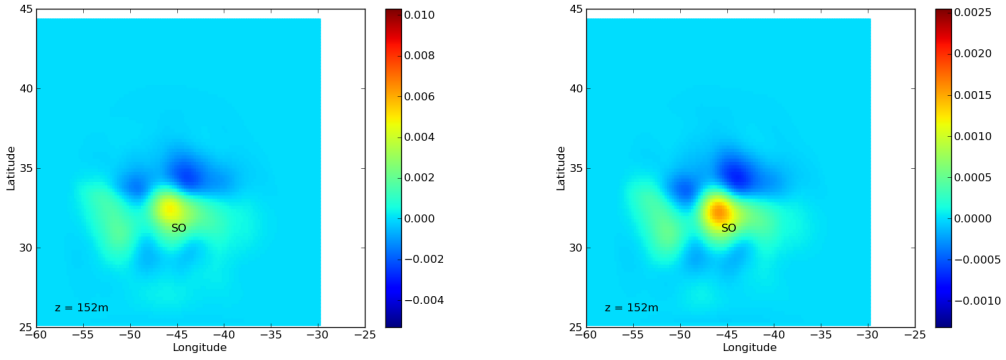
(e) Temperature z=152m

(f) Temperature z=459m

Figure 11: Increment of analysis for u, v, t fields for experiment variant 3 (bias of 1°C, relative error 7.5%), where SO annotation shows the localization of the Single Observation. The colobar palet, dynamic and range are chosen equal to figure 7 for comparison.

3.3.3 SEEK with localization: single observation experiment

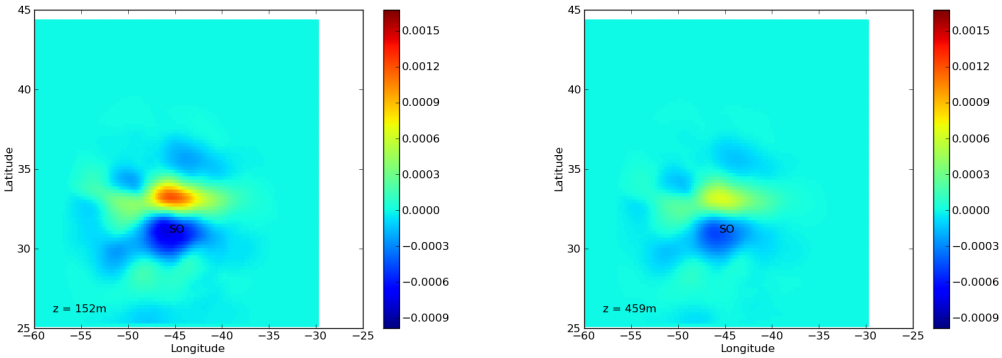
Unless to construct another ensemble, the classical way to handle spurious spatial correlations is to apply a localization process. To illustrate the localization action, we proposed a second demo experiment called *SQB-SEEK/seq_asm_sgl_lroa*. The demo parameters are set identically to the experiment variant 1 of the previous section 3.3.2. The only difference consists in the replacement of the call to the `groa` SESAM module by the call to the `lroa` module (see parameter `ana_mode` in section 4.1.2).



(a) Temperature z=152m

(b) Sea Surface Height z=152m

Figure 12: Increment of analysis of Sea Surface Height and temperature for experiment 2 (bias of 1°C, relative error 5%, with localization), where SO annotation shows the longitude, latitude of the Single Observation. The colobar palet, dynamic and range are chosen equal to figure 6 for comparison.



(a) Zonal speed z=152m

(b) Zonal speed z=459m

(c) Meridonal speed z=152m

(d) Meridional speed z=459m

(e) Temperature z=152m

(f) Temperature z=459m

Figure 13: Increment of analysis for u, v, t fields for experiment 2 (bias of 1°C, relative error 5%, with localization), where SO annotation shows the longitude, latitude of the Single Observation. The colobar palet, dynamic and range are chosen equal to figure 7 for comparison.

4 Designing new experiments

In this section, we explain how to build and launch user-defined SEEK experiments. They can be of the four following types: *trjonly*, *rrkonly*, *trj2rrk* and *seq_asm* (see table 3).

SEEK user-defined experiments	**Experiment type**
trjonly	Ensemble of state vectors
rrkonly	PCA analysis
trj2rrk	Ensemble + PCA analysis
seq_asm	SEEK cycles

Table 3: Types of user-defined experiment.

The directory UTIL/python/exp/SQB-SEEK/user_def is drawn up for user-defined experiments (one can also create his own new directory directly under UTIL/python/exp).

4.1 Experiment parameters: understanding the organization

4.1.1 Configuration and experiment directories

Information related to the experiment definition and parameters is organized in two directories:

- UTIL/python/config:
  this directory stores several subdirectories whose names refer to different ocean grid configurations, and which contains the related NEMO namelist. For instance, UTIL/python/config refers to the SQB grid configuration and stores the corresponding NEMO namelist *namelist_nemo*.

- UTIL/python/exp:
  this directory stores several subdirectories whose names refer to specific demo or user-defined experiments, and which contains several files to specify or modify NEMO and SESAM experiment parameters.

Most of the time, the design of a new DA experiment (variational or sequential) only involves changes in the UTIL/python/exp directory. The subdirectories in UTIL/python/config should not be modified unless the user wants to apply a new ocean configuration.

4.1.2 Modifying the experiment directory *exp*

The demo subdirectories *trjonly*, *rrkonly*, *trj2rrk* and *seq_asm* in UTIL/python/exp/SQB-SEEK provide examples to understand what are the required experiment files.

Any subdirectories of UTIL/python/exp/SQB-SEEK and more generally any demo or user-defined experiment directory, must include a file called *descr*. In the *descr* file, one

can set experiment parameters that are common to SESAM and NEMO (e.g., user working directories, time variables, etc.).

When the type of experiment involves steps with one or several NEMO runs (such as *trjonly*, *trj2rrk* or *seq_asm* experiments), a NEMO namelist-like file is required. It is a short quote of the main NEMO namelist, where are only specified the namelist block parameters that must be modified in comparison with the reference namelist.

Other files related to SESAM are needed: the *sesamlist* configuration file as well as the *algozone* file if localization is applied. Details on those files are provided in appendix B.1 section C.

The *descr* experiment file The demo *descr* file corresponding to the SEEK cycle experiment type is provided below. Some explanations about the parameters are given in table 4 for parameters common to NEMO/SESAM, in table 5 for parameters related to NEMO specific operations and in table 6 for parameters related to SESAM specific operations.

```
1  #================================================================
2  #
3  #  Description file for NEMO/SESAM Experiments
4  #
5  #================================================================
6  #
7  #=================================================
8  # Parameters common to SESAM and NEMO
9  #=================================================
10 tdir = /home/bene/MYRUN/RUNVODA-seq_asm_lroa
11 save_dir = /home/bene/MYRUN/RUNVODA-seq_asm_lroa/save
12 expnam = seq_asm
13 grid = SQB
14 nasmcycl = 2
15 ncycl = 2
16 ndays = 5
17 compile = linux_ifort_bene
18 #
19 #=================================================
20 # NEMO parameters
21 #=================================================
22 date = 20060101
23 tstep = 900
24 nittrjfrq = 96
25 nstock = 96
26 nwrite = 0
```

```
27 outfreq = -1
28 npx = 1
29 npy = 1
30 debug=no
31 #-------------
32 # RESTART
33 #-------------
34 rst_ini = /home/bene/DATA/RST/restart_init.nc
35 #
36 #-----------------
37 # OBSERVATION
38 #-----------------
39 # for seq_asm
40 obs_dirini = /home/bene/DATA/nemov3_testdata/ORCA2_Z31_20060101_monthly
41 obs_namin = EN3_v1c_2006.01.nc, EN3_v1c_2006.02.nc
42 #
43 #====================================================
44 # SESAM parameters
45 #====================================================
46 #-------------------------
47 # ENSEMBLE OF STATE VECTORS
48 #-------------------------
49 # for rrkonly:
50 ens_dirin = /home/bene/DATA/SESAM/TRJ
51 #
52 #-------------------
53 # SESAM EOF - rank red.
54 #-------------------
55 # for trj2rrk or rrkonly:
56 #rrk_num = rrk0005.nc.bas
57 rrk_dirin = /home/bene/DATA/SESAM/RRK/rrk0031.nc.bas
58 #
59 #-------------------
60 # SESAM analysis step
61 #-------------------
62 ana_mode = lroa
63 myobs=tn
64 sgl = true
65 rlon=-45.5
66 rlat=31.
67 dlon=3.
68 dlat=3.
69 rlev=160.
```

```
70 dlev=9.71
71 errel=0.05
72 twinexp = true
73 twinbias = 1.
74 #
75 #-------------------
76 # MASK - GRID for sesam
77 #-------------------
78 # for rrkonly or seq_asm: 2 options for sesam mask/grid file
79 # (i)  already built --> fill up the field grdmsk_filini with path and file name
80 # (ii) to be built   --> fill up the field grdmsk_trjini with path and file name
81 grdmsk_filini = /home/bene/DATA/SESAM/GRDMSK/mskgrd.nc
82 #grdmsk_trjini = /home/bene/DATA/SESAM/TRJ0/tam_trajectory_
```

***descr* parameters**	**Meaning**
tdir	Output directory of the current cycle (see nasmcycl, ncycl)
save_dir	Directory where are archived the outputs of every ended cycle
expnam	SEEK experiment type: *trjonly*, *trj2rrk*, *rrkonly* or *seq_asm*
grid	Ocean configurations
nasmcycl	Number of assimilation cycles (a)
ncycl	Number of cycles (b)
ndays	Number of days per cycles
compile	name of the directory with the fcm file that configures the building of the sources

Table 4: The *descr*: parameters common to NEMO and SESAM. (a) nasmcycl parameter must be defined if experiment type is *seq_asm*. (b) ncycl parameter must be defined if experiment is of the type *trjonly* or *trj2rrk* or if an OAR batch submission is requested.

***descr* parameters**	**Meaning**
date	Date at the beginning of the experiment
tstep nittrjfrq	NEMO time step (seconds) Frequency in time step to save state vector files
nstock	Frequency in time step to save restart file
nwrite	Main frequency for writing down diagnostic files
outfreq	Defines the unit for nwrite parameter
npx, npy	Domain partition for parallel computing
debug	Run in debug mode
rst_ini	Directory where to find the first restart file
obs_dirini	Directory where to find input observation files
obs_namin	Names of the input observation files

Table 5: The *descr* file: parameters related to NEMO operations.

***descr* parameters**	**Meaning**
ens_dirin	For *rrkonly*: directory where are stored ensemble of state vectors (*)
rrk_dirin	For *trj2rrk*, *rrkonly* and *seq_asm*: directory for the reduced rank error covariance matrix (a)
grdmsk_filini	For *rrkonly* and *seq_asm*: path to an already prepared grid and mask file for SESAM
ana_mode	If set to groa, no localization else if set to lroa, localization applied
sgl	Set to *true* = single observation ; else an ENACT temperature profile is applied
twinexp	Set to *true* = twin experiment mode; observations are model values with an applied bias set by the twinbias parameter
twibias	Set the bias on the model equivalent value for twin experiments. If equal to None bias=1
myobs	Select the observed variable (only temperature has been tested)
rlon, rlat, rlev	Select real or single observation position
drlon, drlat	In case of assimilation of a real observation, set the size of the area to search in the database

Table 6: The *descr* file: parameters related to SESAM operations. (a),(b) see appendix B.1 section C for more details.

The NEMO namelist-like files In the case of SEEK experiments, three types of NEMO namelist-like files are defined:

- *namelist_nemo_trj* required for experiments of the type *trjonly* and *trj2rrk*, where NEMO performs the construction of the state vector ensemble;
- *namelist_nemo_init* required for experiments of the type *seq_asm*, where NEMO initializes a given seek cycle and computes the model equivalent to observation;
- *namelist_nemo_frcst* also required for experiments of the type *seq_asm*, where NEMO applies δx_k^a to the background x_k^b.

Experiment namelist *namelist_nemo_trj* It is needed for experiment of type *t*rjonly and *t*rj2rrk, quoted below and followed by explanations:

```
1 &namrun
2    ln_rstart=.true.
3 &namtam
4    ln_trjwri=.true.
5 &nam_asminc
6    ln_bkgwri=.false.
```

Turning on the *ln_trjwri* parameter in block *namtam* (line 4) results in the writting of the netCDF files *tam_trajectory_*.nc*, which are states along the model trajectory used by SESAM to build the ensemble of state vectors. The *ln_rstart* parameter (line 2) must be turned to *true* to start the NEMO simulation from the restart file specified in the *descr* file by parameter *rst_ini* (see table 5). The ensemble can be built in several loops by setting the *ncycl* parameter in the *descr* file (see table 4).

Experiment namelist *namelist_nemo_init* It is needed for experiment of type *s*eq_asm, quoted below and followed by explanations:

```
1 &namrun
2    ln_rstart = .true.
3 &namtam
4   ln_trjwri  = .true.
5 &nam_asminc
6    ln_bkgwri = .true.
7 &namobs
8    ln_ena    = .true.
9    ln_t3d    = .true.
10   ln_s3d    = .true.
11   ln_s_at_t = .true.
```

The *ln_trjwri* parameter is turned on to provide the background state required for the SESAM analysis step. In block *nam_asminc*, the *ln_bkgwri* parameter is turned to *true* to provide the background file *assim_background_state_*.nc* which will be combined to the increment file at the beginning of the NEMO forecast step. Parameters changed in the *namobs* block enable to control the NEMO observation operator capacities: (i) observation extraction from databases, (ii) computation of the model equivalent and (iii) storage of y_k^o and $H\,x_k^f$ in a feedback file format. In our demo case, temperature profiles stored in ENACT files are assimilated and table 7 explains the meaning of the related parameters *ln_ena*, *ln_t3d* and *ln_s_at_t*. The names of the ENACT files must be set in the *descr* file with parameters *obs_dirini* and *obs_namin*.

***namobs* block parameters**	**Meaning**
ln_ena	Logical switch for the ENACT data set
ln_t3d	Logical switch for temperature profiles
ln_s_at_t	Logical switch to compute model S at T observations

Table 7: Parameters of the *namobs* block in the *namelist_nemo_init* file which control the observation extraction and model equivalent calculation.

Experiment namelist *namelist_nemo_frcst* It is needed for experiment of type *s*eq_asm, quoted below and followed by explanations:

```
1 &namrun
2    ln_rstart=.true.
3 &nam_asminc
4    ln_asmdin=.true.
5    ln_trainc=.true.
6    ln_dyninc=.true.
7    ln_sshinc=.true.
8    ln_bkgwri=.false.
9 &namtam
10    ln_trjwri=.true.
```

The *ln_bkgwri* parameter is turned to *false* to prevent the background file *assim_background_state_*.nc* to be overwritten before being combined to the analysis increment (NEMO). To apply the dynamic, the tracer and the see surface height increment of analysis, *ln_dyninc*, *ln_trainc* and *ln_sshinc* must be turned to *true*. Here we apply the increment on the firts time step of the assimilation window as specified with the *ln_asmdin* parameter turned to *true*. To apply the increment according to the IAU technique, *ln_asmdin* must be turned to *false* while *ln_asmdiau* must be turned to *true*.

5 Conclusion

The developments have been made according to VODA task 3.3. The NEMO/NEMOVAR structure is indeed flexible enough to host a sequential DA tool and the SESAM code and executable can now be compiled/ran within the NEMO/NEMOVAR framework. A python driver has been developed to launch SEEK experiments in this framework. Demo experiments are available and some preliminary results are shown. One might now go further and add other schemes of data assimilation in the NEMO/NEMOVAR framework. This work is a step to settle a benchmark for data assimilation experiments where the comparison of sequential, variational and hybrid methods might be performed. This work participates to the anticipation of future needs for a wider panel of DA tools dedicated to the NEMO model.

Appendices

First Appendix

A Initialization of the background error covariance matrix: PCA analysis

A.1 Data matrix $\mathbf{X}^T$

The data matrix $\mathbf{X}^T$ is the general term which designates an m by n matrix, where each of the rows are the repetition of an experiment, i.e., the j^{th} row corresponds to the j^{th} result of a probe, which is defined by n scalar outputs. In the framework of oceanography, a data matrix can be built with an ensemble of m state vectors. These state vectors can be for instance picked along the trajectory at time t_j. In this case, the j^{th} row of the data matrix is the state vector ${X_j}^T$:

$$ {X_j}^T = \big(X(r_1,t_j),\ldots,X(r_i,t_j),\ldots,X(r_n,t_j)\big) \tag{7} $$

where r_i for $i = 1,\ldots,n$ designates the coordinate of the discretized grid of the space domain. In this case, the components of the data matrix are $\left[\mathbf{X}^T\right]_{j,i} = X(r_i,t_j)$.

In the statistical sense, applying a PCA analysis consists in computing the principal modes of variance of the ensemble of m vectors X_j. The data matrix must obviously be of zero empirical mean:

$$ \begin{aligned} E\left[X(r_i,t)\right] &= \textstyle\sum_{j=1}^{m} X(r,t_j) \\ &= 0 \end{aligned} \tag{8} $$

The connexion with error covariance matrices is straightforward. Two empirical error covariance matrices can be built from the data matrix: (i) $\mathbf{X}\mathbf{X}^T$ which size is $n \times n$, and (ii) $\mathbf{X}^T\mathbf{X}$ which size is $m \times m$. Both matrices may have a statistical sense, but this sense may

differ. When applied to the ensemble of state vectors, it is easy to show that $\mathbf{X}\mathbf{X}^T$ estimates spatial covariances on the basis of a time sampled ensemble:

$$\begin{aligned} \mathbf{Cov}\left(X(r,t_0),X(r',t_0)\right) &= \mathbf{E}\left[X(r,t_0),X(r',t_0)\right] \simeq [\mathbf{X}\mathbf{X}^T]_{i,j} \\ &= \sum_{p=1}^{m} X(r_i,t_p)\,X(r_j,t_p) \end{aligned} \tag{9}$$

On the other hand, $\mathbf{X}^T\mathbf{X}$ estimates temporal covariances on the basis of a space sampled ensemble:

$$\begin{aligned} \mathbf{Cov}\left(X(r_0,t),X(r_0,t')\right) &= \mathbf{E}\left[X(r_o,t),X(r_o,t')\right] \simeq [\mathbf{X}^T\mathbf{X}]_{i,j} \\ &= \sum_{p=1}^{n} X(r_p,t_j)\,X(r_p,t_j) \end{aligned} \tag{10}$$

Both $\mathbf{X}\mathbf{X}^T$ and $\mathbf{X}^T\mathbf{X}$ are squared, symmetric and positive. An eigen mode decomposition can be applied and eigenvalues are positive or zero. Both endomorphisms ($\mathbf{X}\mathbf{X}^T : E_n \to E_n$ and $\mathbf{X}^T\mathbf{X} : F_m \to F_m$, where E_n and F_m are linear spaces) have the same rank, which is the rank of $\mathbf{X}$ (or $\mathbf{X}^T$), let us say r. They though have both r strictly positive eigenvalues.

If we designate by ${\sigma_i}^2$ for $i = 1,\ldots,r$ the eigenvalues of $\mathbf{X}\mathbf{X}^T$ and v_i the corresponding linearly independent set of r orthonormal eigenvectors, and if we build a new set of r vectors u_i such as:

$$u_i = \mathbf{X}^T \frac{v_i}{\sigma_i} \tag{11}$$

then we have:

$$\begin{aligned} \mathbf{X}^T\mathbf{X}u_i &= \mathbf{X}^T \quad \mathbf{X}\mathbf{X}^T \tfrac{v_i}{\sigma_i} \\ &= \mathbf{X}^T \quad \tfrac{{\sigma_i}^2 v_i}{\sigma_i} \\ &= \sigma_i\,\mathbf{X}^T v_i \\ &= {\sigma_i}^2\,u_i \end{aligned} \tag{12}$$

This shows that both endomorphisms $\mathbf{X}\mathbf{X}^T$ and $\mathbf{X}^T\mathbf{X}$ have moreover the same eigenvalues ${\sigma_i}^2$, but two different set of eigenvectors, which are v_i for the former, and u_i for the later, $\forall i = 1,\ldots,r$. The two basis are related by the relationship 11 or equivalently:

$$v_i = \mathbf{X}\frac{u_i}{\sigma_i} \tag{13}$$

To synthesize, we therefore have the following equations:

$$\begin{aligned} \mathbf{X}\mathbf{X}^T v_i &= {\sigma_i}^2\,v_i \\ \mathbf{X}^T v_i &= \sigma_i\,u_i \\ \mathbf{X}^T\mathbf{X}u_i &= {\sigma_i}^2\,u_i \\ \mathbf{X}u_i &= \sigma_i\,v_i \end{aligned} \tag{14}$$

We can build the two vector basis $\{v_i\}_{i=1,n}$ and $\{u_i\}_{i=1,m}$, spanning E_n and F_m by adding to the two sets of r eigenvectors $\{v_i\}_{i=1,r}$ and $\{u_i\}_{i=1,r}$ respectively: (i) $n-r$ orthonormal

vectors v_i^o spanning $\ker \mathbf{X}^T$ (or equivalently $\ker \mathbf{X}\mathbf{X}^T$), for the former, and (ii) $m-r$ orthonormal vectors v_i^o spanning $\ker \mathbf{X}$ (or equivalently $\ker \mathbf{X}^T\mathbf{X}$), for the later. We then introduce the two unitary matrices V and U, whose columns are vectors $v_i \forall i = 1, \ldots, n$ and $u_i \forall i = 1, \ldots, m$, to apply a change of basis from the initial basis to the $\{v_i\}_{i=1,n}$ and $\{u_i\}_{i=1,m}$ eigen basis. Then we get the singular decomposition of either matrix $\mathbf{X}$ or matrix $\mathbf{X}^T$:

$$\mathbf{X} = \mathbf{U}\,\mathbf{\Sigma}\,\mathbf{V}^T \tag{15a}$$

$$\mathbf{X}^T = \mathbf{V}\,\mathbf{\Sigma}^T\,\mathbf{U}^T \tag{15b}$$

where $\mathbf{\Sigma}$ is an $n \times m$ matrix. It has zero everywhere except on its diagonal where are the r square roots of the eigenvalues ${\sigma_i}^2$. This is straightforward when transforming equations 15a and 15b as below:

$$\begin{aligned} \mathbf{X}\mathbf{X}^T &= \mathbf{U}\,\mathbf{\Sigma}\,\mathbf{V}^T\,\mathbf{V}\,\mathbf{\Sigma}^T\mathbf{U}^T \\ &= \mathbf{U}\,\mathbf{\Sigma}\mathbf{\Sigma}^T\mathbf{U}^T \end{aligned} \tag{16}$$

$$\begin{aligned} \mathbf{X}^T\mathbf{X} &= \mathbf{V}\,\mathbf{\Sigma}^T\,\mathbf{U}^T\,\mathbf{U}\,\mathbf{\Sigma}\mathbf{V}^T \\ &= \mathbf{V}\,\mathbf{\Sigma}^T\mathbf{\Sigma}\mathbf{V}^T \end{aligned} \tag{17}$$

In conclusion, the PCA analysis is in fact equivalent to perform a singular decomposition, either of the data matrix or of its transpose, the $n \times m$ matrix $\mathbf{X}$ or of (the transpose of the data matrix).

B Complement to Kalman equations

B.1 Model error representation

Let us denote:

- the background state vector x_0^f,
- the forecast state vector x_k^f,
- the observation vector y_k^o at time t_k,
- the non linear dynamical and observation models $\mathcal{M}$, $\mathcal{H}$,
- the linear tangent operators $\mathbf{M}_{k,k+1}$, $\mathbf{H}_k$,
- and the "true" but unknown state and observation vector x_k^t and y_k^t.

Error covariance matrices of DA system are based on the definition of errors between the "true" and estimated vectors of the problem [4]:

[4] This formalism relies on normal or lognormal error statistics.

- The state vector x_k^f at time t_k is the sum of the "true" vector x_k^t and the background error ϵ_k^f:

$$x_k^f = x_k^t + \epsilon_k^f \tag{18}$$

- The model applied to x_k^t the "true" state vector at time t_k, provides x_{k+1}^t the "true" state vector at time t_{k+1} within the model error η_k (which is due to unresolved physics):

$$x_{k+1}^t = \mathcal{M}_{k,k+1}(x_k^t) + \eta_k \tag{19}$$

- From the "true" state vector x_k^t one can predict the model equivalent to the "true" observation y_k^t within the observation error ϵ_k^o:

$$y_k^t = \mathcal{H}_k(x_k^t) + \epsilon_k^o \tag{20}$$

The error covariance matrices of background and/or forecast P_k^f, of model Q_k and of observation R_k are:

$$\begin{array}{ll} E[\epsilon_k^f \, \epsilon_k^{f\,T}] & = P_k^f \\ E[\epsilon_k^o \, \epsilon_k^{o\,T}] & = R_k \\ E[\eta_k \, \eta_k^{\,T}] & = Q_k \end{array} \tag{21}$$

The SEEK equations rely on the following hypothesis regarding errors:

- No bias on:
  - the observation error
$$E[\epsilon_k^o] = 0 \tag{22}$$
  - the dynamic model error
$$E[\eta_k] = 0 \tag{23}$$
- No time correlation between:
  - observation errors
$$E[\epsilon_k^o \, \epsilon_{k'}^o] = 0 \quad \forall \, k \neq k' \tag{24}$$
  - dynamic model errors
$$E[\eta_k \, \eta_{k'}] = 0 \quad \forall \, k \neq k' \tag{25}$$
- No correlation between dynamic model and observation:

$$E[\eta_k \, \epsilon_{k'}^o] = 0 \quad \forall \, k, k' \tag{26}$$

Second Appendix

C More on SESAM

C.1 State vector and error covariance matrices

C.1.1 Storage with a common file format

SESAM applies a Square Root approach regarding error covariance matrices of background, forecast or analysis, denoted B, P^f and P^a. The square root matrices are defined and handled as ensemble of vectors. To perform an EOF analysis and initialize the background error covariance matrix B, an ensemble of states along the model trajectory must be provided under the form of input files. The Principal Component Analysis (PCA, or equivalently EOF) enables a reduction of order of B, by picking up the directions of greatest variance. Because these directions are simply orthogonal and normalized state vectors, covariance matrices are stored in the same type of file as state vectors themselves.

C.1.2 Structured and unstructured file format

SESAM makes use of two file structures of binary format to store state vectors. Some SESAM operations handles one structure rather then the other. The first one is the model file format which provides data fields defined on the model grid, while the second is free from model grid information and reorganizes the fields in a single vector. The later file structure is a direct copy of the code internal variables.

C.1.3 Full state vector or observed section

The state vector and its error covariance matrix are referred to as V_x and C_x in the SESAM documentation. When preparing an assimilation experiment, the user must define by advance which section of the state vector will be observed. To this section is associated a vector object called V_y. The related error covariance matrix is denoted C_y. These intermediate objects have several purposes: reduction of the size of the loaded data, twin experiments ... etc. SESAM constructs the V_x and V_y objects out of the NEMO output files. It requires information on the structure of the NEMO output files. This information is stored in the *sesamlist*, which is a parameter file (see section C.4 of the appendix).

C.2 Observations, error covariance and observation operators

SESAM can read several databases types. Databases objects are denoted I_o. In this work, we applied the simplest type of databases which is made of ASCII files. ASCII files must have the extension *.adbs* and a well-defined structure which is described in section C.4.1. SESAM enables to assimilate observations that are measurements of the model variables (for instance, a temperature profile). In the SESAM documentation, observation objects are designated by V_o and represent the data file after the databases extraction. The observation error covariance matrices R are denoted C_o in SESAM. The default option for corresponds to a diagonal with unit variance R matrix. C_o objects may however be built on the basis of an ensemble of observations.

C.3 Input and output files or directory: naming conventions

SESAM I/O files obey specific naming conventions that we address hereafter. Any vector object, V_x, V_y or V_o requires a single file to be defined, while any error covariance matrix objects C_x, C_y or C_o, require several files. To each of these objects correspond structured (grid dependent) or unstructured binary files (grid independent). To adapt to different model settings, SESAM takes in input several structured binary file formats. In the present work, we only handled the generic NetCDF format which conforms to the trajectory output format of NEMO version 3.0. Specific naming conventions must be observed so that SESAM recognizes the objects to input or output. For instance, in the case of a NetCDF file storing a state of the model trajectory, the extension of the file must be *.nc*, while if it only stores the observed section, its extension ought to be *.ncdta*. If the state of the model trajectory is converted into the specific SESAM unstructured binary file, the file extension must be *.cpak*. The corresponding SESAM unstructured binary file for V_o objects must have extension *.cobs* (more details on V_o objects are provided in section C.4.1 of the appendix). Table 8 summarizes the file naming rules.

Vector object	Related stem	Binary formatted file extension	Binary unformatted file extension
Vx	var	.nc	.cpak
Vy	dta	.ncdta	
Vo	obs		.cobs

Table 8: SESAM vector objects and naming rules

Regarding the matrix objects (i.e.: ensemble of vector directions and amplitudes), the naming conventions target both the vector files and their directory :

- `namejjjj.ext.bas`
- `vctjjjj.ext`

where:

- `name` : a user defined name,
- `jjjj` : a 4 digits number $r+1$, where r defines the rank reduction,
- `ext` : extension which defines the type of vector objects stored according to Table 8.

C.4 Configuration files

The model output fields are stored in binary files that are structured on a grid, with potentially a mask to define land sub-domains. SESAM being independent from the model it applies on, template and configuration files are necessary.

Sesamlist The sesamlist is a configuration file where must be recorded the names of the state vector variables, their dimension, the fact that they are likely to be observed or not. Information on the grid (its type regular or irregular) and on the mask must also be registered. Two types of NEMO template files must be provided: the grid and mask files, on one side for the state vector and on the other side for its observed section. Those template files are needed to convert NEMO output files in V_x and V_y formats, and vice versa. The sesamlist applied for the demo is quoted (see lines 38 and 47 for mask files, and lines 56 and 60 for grid files associated to V_x and V_y respectively):

```
1   NPRINT=4
2   ########################################
3   #STATE VECTOR Vx and OBSERVED SECTION FLAG
4   ########################################
5   VAREND=5
6   # temperature
7   VAR_NAM-1=tn
8   VAR_DIM-1=3
9   DTA_ACT-1=.TRUE.
10  # zonal speed
11  VAR_NAM-2=un
12  VAR_DIM-2=3
13  DTA_ACT-2=.FALSE.
14  # meridian speed
15  VAR_NAM-3=vn
16  VAR_DIM-3=3
17  DTA_ACT-3=.FALSE.
18  # salinity
19  VAR_NAM-4=sn
20  VAR_DIM-4=3
21  DTA_ACT-4=.FALSE.
22  # sea surface height
23  VAR_NAM-5=sshn
24  VAR_DIM-5=2
25  DTA_ACT-5=.FALSE.
26  ###########################
27  #OBSERVED SECTION VECTOR Vy
28  ###########################
29  # temperature only
30  DTA_NAM-1=tn
31  DTA_DIM-1=3
32  #################
33  # MASK DEFINITION
34  #################
```

```
35  # Default: Mask file valid for all variables
36  # Vx - VAREMSK=4 for .nc files
37  VAREMSK=4
38  VARFMSK=mskgrd.nc
39  VARVMSK=999999.
40  VARDMSK-1=3
41  VARDMSK-2=3
42  VARDMSK-3=3
43  VARDMSK-4=3
44  VARDMSK-5=2
45  # Vy
46  DTAEMSK=4
47  DTAFMSK=mskgrd.ncdta
48  DTAVMSK=999999.
49  DTDAMSK-1=3
50  ##################
51  # NEMO MODEL GRID
52  ##################
53  # Vx - VAREMSK=4 for .nc files and VARNGRD=3 irregular grid Lat,Lon(i,j)
54  VAREGRD=4
55  VARNGRD=3
56  VARFGRD=mskgrd.nc
57  # Vy
58  DTAEGRD=4
59  DTANGRD=3
60  DTAFGRD-1=mskgrd.ncdta
61  ######################################################################
62  #Additional configuration for the generic NetCDF format (.nc): Vx
63  ######################################################################
64  # dimensions and grid variables
65  VARXDIM=x
66  VARYDIM=y
67  VARZDIM=z
68  VARTDIM=t
69  VARXNAM=nav_lon
70  VARYNAM=nav_lat
71  # temperature
72  VARIFIL-1=tn
73  VAROFIL-1=tn
74  VARIPOS-1=51
75  VAROPOS-1=51
76  VARINAM-1=all
77  VARONAM-1=all
```

```
78  # zonal speed
79  VARIFIL-2=un
80  VAROFIL-2=un
81  VARIPOS-2=52
82  VAROPOS-2=52
83  VARINAM-2=all
84  VARONAM-2=all
85  # meridional speed
86  VARIFIL-3=vn
87  VAROFIL-3=vn
88  VARIPOS-3=53
89  VAROPOS-3=53
90  VARINAM-3=all
91  VARONAM-3=all
92 # salinity
93  VARIFIL-4=sn
94  VAROFIL-4=sn
95  VARIPOS-4=54
96  VAROPOS-4=54
97  VARINAM-4=all
98  VARONAM-4=all
99 # sea surface height
101 VARIFIL-5=sshn
102 VAROFIL-5=sshn
103 VARIPOS-5=55
104 VAROPOS-5=55
105 VARINAM-5=all
106 VARONAM-5=all
107 ###############################################################
108 #Additional configuration for the generic NetCDF format (.nc): Vy
109 ###############################################################
110 # dimensions and grid variables
111 DTAXDIM=x
112 DTAYDIM=y
113 DTAZDIM=z
114 DTATDIM=t
115 DTAXNAM=nav_lon
116 DTAYNAM=nav_lat
117 DTAZNAM=z
110 # temperature
111 DTAIFIL-1=tn
112 DTAOFIL-1=tn
113 DTAIPOS-1=11
```

```
114 DTAOPOS-1=11
115 DTAINAM-1=all
116 DTAONAM-1=all
117 ################
118 # OBSERVATION Vo
119 ################
120 OBSNDBS-1=1
121 OBS_NAM-1:1=POT
122 OBS_DIM-1:1=3
123 OBSINAM-1:1=POT
```

Localization parameter file Here is an excerpt of an algozone.cfg file:

```
1 1
46 46
18. 18.
1
tn 1
un 1
vn 1
sn 1
sshn 1
1
1.0 0.0
tn 1
```

C.4.1 Databases

ASCII type Databases of ASCII type must have the extension *.adbs*. The first line must contain the number of observations. The next lines provide the data themselves with the following structure:

```
longitude (°)  latitude (°)  depth (m)  time (julian days since 1950)  observation
```

Before any treatment, this data file is converted in a SESAM V_o object and stored in a *.cobs* file (see table 8). The extraction step consists in localizing the observed model variable on the state vector grid and then to calculate the corresponding components of H, the observation operator. To one observation is attached 8 components of H, that are the interpolation coefficients on the structured grid of the domain. The header of the binary data file is shown below:

```
dimensions:
obsidx = 48 ;
itpidx = 8 ;
```

```
dta = 1 ;
namelength = 5 ;
variables:
char name(dta, namelength) ;
int dim(dta) ;
int nbr(dta) ;
float lon(obsidx) ;
float lat(obsidx) ;
float depth(obsidx) ;
float POT(obsidx) ;
float itp_pos(itpidx, obsidx) ;
float itp_coef(itpidx, obsidx) ;
```

C.5 SESAM diagnostic files

Each time a SESAM module is called, an ASCII file called `sesam.output` is recorded in the working directory where important results can be found. See appendix D.1.

D More on the python script

D.1 Saving SESAM diagnostic files

The python script `seq_assim_driver.py` preserves the `sesam.output` files in the saving directory. They are renamed according to the SESAM action applied. For instance, the file named `sesam_groa.output` is saved after the analysis step with the groa module. Here is an excerpt of this sesam diagnostic file, where one can check for instance the analysis increment coefficients on the principal component basis (pay attention that the orthogonal vector columns of the S_0^f matrix are not of unit length, but of the length of the principal component standard deviation).

```
ALGOROA_U: computing the ROA coefficients (c)
 -----------------------------------------
 ==> ALGOROA_U: Xa-ROA coefficients
 array : coefrz
 -----------------------------------------
 coefrz(           2 )=  3.492172104935640E-003
 coefrz(           3 )= -0.106457321305230
 coefrz(           4 )= -0.287005321176942
 coefrz(           5 )= -7.298499062401952E-002
 coefrz(           6 )= -1.732570307991218E-002
 coefrz(           7 )=  5.423290027544080E-002
 coefrz(           8 )= -3.856688659280162E-002
 coefrz(           9 )= -1.435348130359445E-003
```

```
 coefrz(          10 )=  2.777369872765285E-002
 coefrz(          11 )= -5.269387747197801E-002
 coefrz(          12 )= -0.146614444269370
 coefrz(          13 )=  9.127680970525288E-002
 coefrz(          14 )= -1.661452063989079E-002
 coefrz(          15 )= -0.228593454575274
 coefrz(          16 )=  9.256611063733509E-002
 coefrz(          17 )=  4.929063066109286E-003
 coefrz(          18 )= -9.685069740965432E-002
 coefrz(          19 )= -8.513948569870358E-002
 coefrz(          20 )= -5.353112441168610E-002
 coefrz(          21 )=  1.178926593509783E-002
 coefrz(          22 )= -9.003216158593319E-002
 coefrz(          23 )= -0.127399574646197
 coefrz(          24 )=  5.594369451034573E-002
 coefrz(          25 )= -4.242851291698317E-003
 coefrz(          26 )= -7.050862321081121E-002
 coefrz(          27 )=  0.142886169325046
 coefrz(          28 )= -9.570423122063972E-003
 coefrz(          29 )= -1.682697620524466E-002
 coefrz(          30 )= -0.115931931293326
 coefrz(          31 )= -1.721373122891485E-002
```

An excerpt of the same file after applying the geof module to compute the PC is also shown below. One can read for instance the standard deviation associated to each principal components:

```
************************
*     routine algortho  *
************************

 ---------------------------------------
 ==> eighenvalues :FROM NORM OF BASIS (METHOD 1)
 ---------------------------------------
  jr | valp | SQRT(valp) | % variance    | total of % variance
2  0.157783614644228E+04   0.397219856810090E+02   0.735511640418145E+02    73.551164
3  0.452755807682481E+03   0.212780593025417E+02   0.211053072632574E+02    94.656471
4  0.684177848120003E+02   0.827150438626495E+01   0.318930943839236E+01    97.845780
5  0.286700770529267E+02   0.535444460732640E+01   0.133646167579950E+01    99.182242
6  0.106892445832421E+02   0.326944101999748E+01   0.498281385933427E+00    99.680523
7  0.486357672750791E+01   0.220535183757783E+01   0.226716652753506E+00    99.907240
8  0.198989860024021E+01   0.141063765731679E+01   0.927595420493167E-01   100.000000
 ---------------------------------------
```

D.2 More on file conversion

Figure 14 provides some information for future developers to identify the files that are converted off-line for the NEMO/SESAM communication during the SEEK cycle.

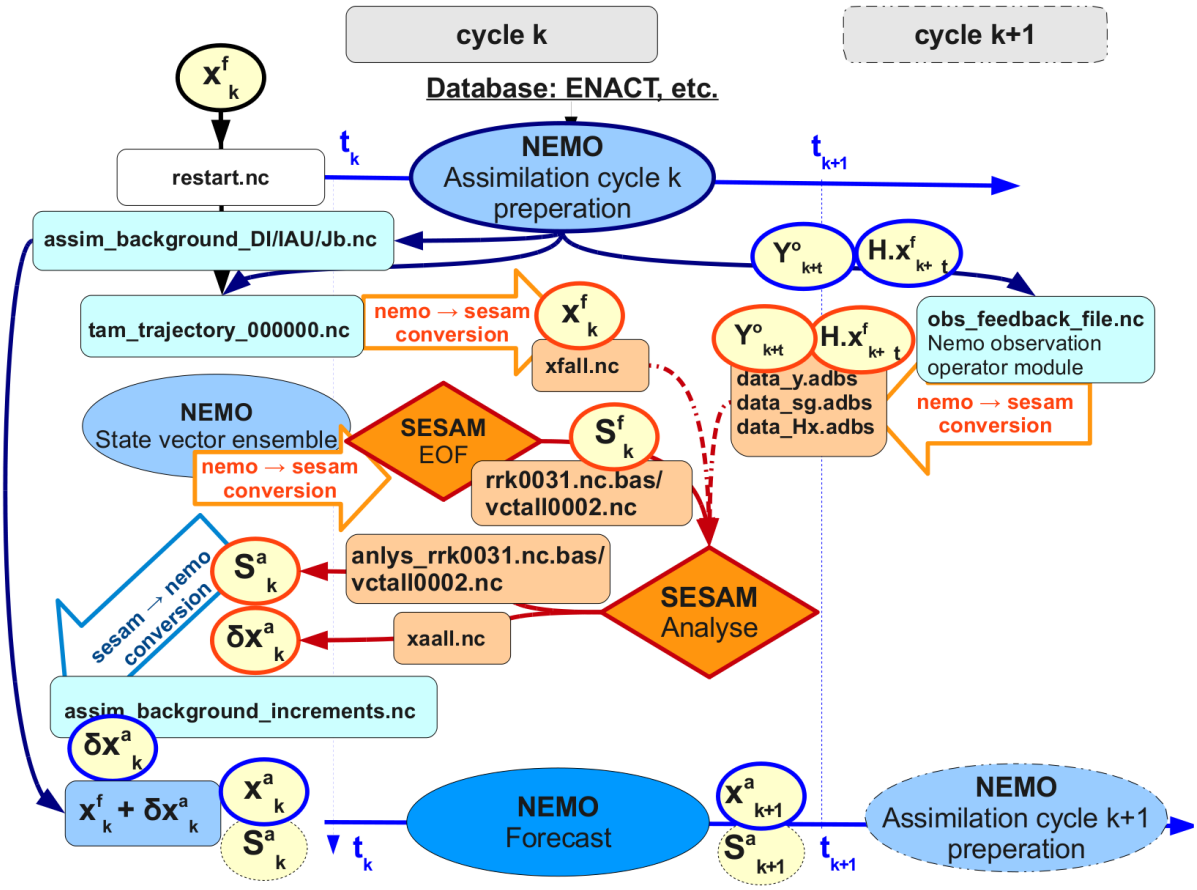


Figure 14: **SEEK cycle operations between NEMO and SESAM** off-line communication between NEMO and SESAM involved the shown files and conversion.